AFRL-VA-WP-TP-2004-313

GRID DISCRETIZATION BASED METHOD FOR ANISOTROPIC SHORTEST PATH PROBLEM OVER CONTINUOUS REGIONS

Zhanfeng Jia
Pravin Varaiya

APRIL 2004

This work, resulting from Department of Air Force contract number F33615-01-C-3150, has been submitted for publication in the Proceedings of the 2004 IEEE Conference on Decision and Control. If published, IEEE may assert copyright. If so, the United States has for itself and others acting on its behalf an unlimited, nonexclusive, irrevocable, paid-up, royalty-free worldwide license to use for its purposes.

AIR VEHICLES DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*
April 2004

2. REPORT TYPE
Conference Paper Preprint

3. DATES COVERED *(From - To)*

4. TITLE AND SUBTITLE
GRID DISCRETIZATION BASED METHOD FOR ANISOTROPIC SHORTEST PATH PROBLEM OVER CONTINUOUS REGIONS

5a. CONTRACT NUMBER
F33615-01-C-3150

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
69199F

6. AUTHOR(S)
Zhanfeng Jia
Pravin Varaiya

5d. PROJECT NUMBER
ARPF

5e. TASK NUMBER
04

5f. WORK UNIT NUMBER
T3

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of California, Berkeley
Electrical Engineering and Computer Science Department
Berkeley, CA 94720

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Vehicles Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson Air Force Base, OH 45433-7542

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/VACA

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-VA-WP-TP-2004-313

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
Conference paper to be presented at the IEEE Conference on Decision and Control (CDC), Paradise Island, Bahamas, 14-17 December 2004. This work, resulting from Department of Air Force contract number F33615-01-C-3150, has been submitted for publication in the Proceedings of the 2004 IEEE Conference on Decision and Control. If published, IEEE may assert copyright. If so, the United States has for itself and others acting on its behalf an unlimited, nonexclusive, irrevocable, paid-up, royalty-free worldwide license to use for its purposes.

14. ABSTRACT
This paper presents a method to find the shortest path between two points over a continuous region. The length of a path is the integral of the cost along the path. The cost can be anisotropic, meaning that it depends on both the position on the path and its direction. The method uses a simple rectangular grid to discretize the region, independent of the cost. Because the cost is anisotropic, rectilinear paths connecting adjacent grid points may not approximate the optimal path. To overcome this "limit-on-direction" problem, the method searches over shifted versions of the rectilinear paths. A Bellman-Ford style algorithm finds the best shifted path. Theoretical analysis and numerical experiments ensure efficiency of the algorithm.

15. SUBJECT TERMS
Anisotropic cost, path planning, grid selection

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
a. REPORT	b. ABSTRACT	c. THIS PAGE			Mark Mears
Unclassified	Unclassified	Unclassified	SAR	14	19b. TELEPHONE NUMBER *(Include Area Code)* (937) 255-8685

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18

Grid Discretization Based Method for Anisotropic Shortest Path Problem over Continuous Regions

Zhanfeng Jia and Pravin Varaiya

Abstract— **The paper presents a method to find the shortest path between two points over a continuous region. The length of a path is the integral of the cost along the path. The cost can be anisotropic, meaning that it depends on both the position on the path and its direction. The method uses a simple rectangular grid to discretize the region, independent of the cost. Because the cost is anisotropic, rectlinear paths connecting adjacent grid points may not approximate the optimal path. To overcome this "limit-on-direction" problem, the method searches over shifted versions of the rectlinear paths. A Bellman-Ford style algorithm finds the best shifted path. Theoretical analysis and numerical experiments ensure efficiency of the algorithm.**

I. Introduction

The problem is to find the shortest path over region $\mathcal{R} \subset \mathbb{R}^n$. For a starting point $x_s \in \mathcal{R}$, the 'length' $V(x,p)$ of a continuous path p from x_s to a destination x is the integral of the cost function $c(x,v)$ along the path p,

$$V(x,p) = \int_0^T c(p(t),\dot{p}(t))dt. \quad (1)$$

The path $p : [0,T] \rightarrow \mathcal{R}$ must be continuously twice differentiable; $p(0) = x_s$, $p(T) = x$; and such that $||\dot{p}|| = 1$, and $||\ddot{p}||$ is bounded. We want a method that finds the best path,

$$V(x) = \min_{p \in P(x_s,x)} V(x,p). \quad (2)$$

$V(x)$ is the *value function* at x. The cost $c : \mathcal{R} \times \mathbb{R}^n \rightarrow (0,\infty)$ depends on both position $p(t)$ and velocity $\dot{p}$. However, as $||\dot{p}|| = 1$, the cost c depends only on position and direction. Problem (2) is called the *Anisotropic Shortest Path* (ASP) problem. The special case, when $c(p,\dot{p})$ is independent of $\dot{p}$, is the *Isotropic Shortest Path* (ISP) problem.

The ASP problem arises naturally in path planning. Our motivation is to find paths for Unmanned Aerial Vehicles (UAVs) that can reach a designated target while minimizing the risk of detection from enemy radar. The radar signature of a UAV is a function of its distance from the radar and its flight angle, so the accumulated risk faced by the UAV along a path can be expressed as in (1).

One approach to ASP problem is to select a uniform grid of points spaced ϵ apart, with a link between *any* two nearby grid points. The cost of a link connecting points x, y is taken to be

$$c(x, \frac{x-y}{\| x-y \|}) \times \| x-y \|, \quad (3)$$

and one can use an algorithm like Dijkstra's to find the shortest path across the graph. Kim and Hespanha [1] show that the shortest path along the graph converges to a shortest continuous path as the graph is refined. For a two-dimensional region, the graph constructed above has $O(\epsilon^{-1})$ vertices and $O(\epsilon^{-2})$ links, which is inefficient.

Kim and Hespanha [1] propose an efficient adaptive sampling scheme that takes into account the characteristics of the cost function over the region. Using a uniform grid is inefficient because of the "limit-on-direction" problem, discussed in the next section.

The proposed method finds approximately shortest paths on a graph with a uniform grid. The crucial difference from the graph described above is that there is a link only between adjacent vertices. Consequently, the graph is easy to construct, and unlike in (3) there is no need to evaluate the distance between pairs of point in order to select the links in a shortest path. The algorithm, called the *Shiftable Segment-by-Segment Path* (SSSP) algorithm, overcomes the limit-on-direction problem by allowing the paths to shift away from the grid points. The shifts are local, so the shifted path is a close variant to the discrete path, and better approximates the desired shortest path.

Section II provides a brief review. Section III introduces the segment-shifting idea and discusses how it helps to find the shortest continuous path.. Section IV describes the SSSP algorithm and analyzes its complexity. Section V presents numerical experiments. Section VI concludes the paper.

II. Isotropic and Anisotropic Shortest Path Problems: Related Work

An efficient Dijkstra-like algorithm for the Isotropic Shortest Path (ISP) problem was first proposed by Tsitsiklis [4] in 1995, and later independently proposed by Adalsteinsson and Sethian [2] and called the *Fast Marching* algorithm. Using a uniform grid, the algorithm mimics a wave front propagation. The idea comes from geometric optics where a wave front propagates at different speeds depending on the medium which, in this case, is inversely proportional to the cost function at each point x.The cost does not depend on the direction $\dot{p}(t)$, so Problem (2)

This work was supported by DARPA-N66001-00-C-8062 and DARPA-F33615-01-C-3150.

Z. Jia and P. Varaiya are with the Electrical Engineering and Computer Science Department, University of California, Berkeley, CA 94720, USA. {jia,varaiya}@eecs.berkeley.edu

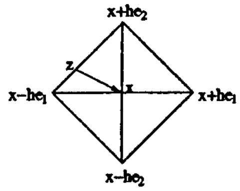

Fig. 1. Value function update in the Fast Marching method: $V(x) = V(z) + c(x) \cdot \text{len}(xz)$, where $V(z)$ is interpolated by the two adjacent points of x, and $\text{len}(xz) = h\sqrt{s^2 + (1-s)^2}$.

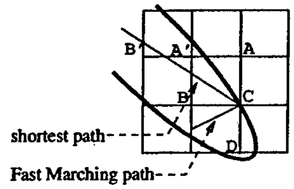

Fig. 2. Failure of the FM method on an anisotropic case with elliptic cost profile. While the shortest path comes from between point A' and B, the result of FM is between point B and D.

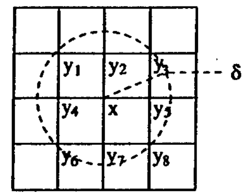

Fig. 3. Limit-on-direction problem of the grid sampling. For a grid discretization of step h, and a neighborhood distance $\delta = \sqrt{2}h$, there are eight neighbor points of x, implying only eight possible directions. The dashed circle illustrates the neighborhood area of point x.

simplifies to

$$V(x) = \min_{p \in P(x_s, x)} \int_0^T c(p(t))dt. \qquad (4)$$

$V(x)$ can be viewed as the time the wave first arrives at x. The Fast Marching algorithm strategy uses the value function to order the selection of grid points, similar to Dijkstra's algorithm, as the current smallest wave front. The underlying theory is the *causality relationship*, which states that the arrival time (value function) $V(x)$ at x depends only on the neighbors with strictly smaller V-values. The proof of this statement can be found in [4].

Using a uniform grid and the causality relationship, the Fast Marching method updates the value function at a grid point x according only to the values of its adjacent grid points. Specifically, in the two dimensional case the update process is given by

$$V(x) = \min_{\alpha \in \mathcal{A}} \min_{0 \leq s \leq 1} \Big[hc(x)\sqrt{s^2 + (1-s)^2} + sV(x + h\alpha_1 e_1) + (1-s)V(x + h\alpha_2 e_2) \Big], \qquad (5)$$

where e_1 and e_2 are the unit vectors in $\mathbb{R}^2$, $\alpha = (\alpha_1, \alpha_2)$ is an element of $\mathcal{A} = \{-1, 1\}^2$, and h represents the discretization step (see Fig. 1). Equation (5) resembles a discrete approximation of the Hamilton-Jacobi (HJ) equation.

The Fast Marching algorithm, however, fails in the anisotropic case, one reason being the failure of the causality relationship on the grid, as illustrated by Sethian and Vladimirsky [3]. In their example, the cost function has an elliptic profile and is uniform across the region. The constant-value contours are concentric ellipses with center x_s, and the shortest path to any point x is the straight line connecting x_s and x.

The Fast Marching algorithm cannot converge to these shortest paths. Suppose we want to update the value function at point C in Fig. 2. The thick elliptic curve is a constant-value contour, so $V(A)$ is larger than $V(C)$, and both $V(B)$ and $V(D)$ are less than $V(C)$. Given the causality relationship, the Fast Marching algorithm uses points B and D to update $V(C)$ and omits the combination of points A and B. Thus, the path suggested by the Fast Marching algorithm can not approximate the optimal continuous path. Modifications of the Fast Marching method have been proposed to address this problem. For example, in [3], the update procedure considers not only the four adjacent points, but non-adjacent points such as A' and B', at the cost of larger time complexity.

Another difficulty introduced by anisotropy comes from the restriction that $||\vec{p}||$ is bounded. The restriction is reasonable in the UAV scenario, in which the UAV's ability to change directions is limited. But the bound leads to a failure of the *Principle of Optimality*, because a partial section of a shortest path may not be itself be a shortest path. Therefore, the value function $V(x)$, as a function of position alone does not propagate properly. One can extend the value function, with both position and direction as state, but this Fast Marching much more complicated.

Alternative approaches to the ASP problem focus on paths instead of value functions. The idea is, first, to reduce the continuous region to a discrete graph, and then to compute the shortest paths across the graph. Path-oriented methods fit the ASP Problem (2) well because paths carry direction information. Moreover, path-oriented methods remove the additional procedure needed to reconstruct paths from the value functions.

The basic step associated with path-oriented methods is sampling, which reduces the region into a discrete graph. The simplest is (uniform) grid sampling. Unfortunately, grid sampling is not efficient in the sense that the number of vertices and edges grows very fast if we want the shortest discrete path to approximate both the position and direction of the shortest continuous path. This is because of the "limit-on-direction" effect.

Remember that the discrete paths are composed of straight segments between sample points. Consider a two-dimensional grid and a small neighborhood distance $\delta =$

$\sqrt{2}h$ as in Fig. 3. The dashed circle centered at x shows the neighborhood of x. Denote the eight grid points in the neighborhood by $\mathcal{Y}_x = \{y_1, ..., y_8\}$. Suppose only these eight points in $\mathcal{Y}_x$ are available to form a segment with x. Then every possible segment in the grid graph has to lie either along grid lines (as from y_2, y_4, y_5, or y_7 to x), or on diagonal lines of grids, (as from y_1, y_3, y_6, or y_8 to x), thus on one of the eight directions. The discrete paths composed of these segments can never approximate a continuous path along (say) a 30 degree line. Increasing the fineness of discretization or the neighborhood distance both introduce more directions, but cannot eliminate the limit-on-direction effect completely. In fact, even the random sampling works more efficiently than grids.

Kim and Hespanha in [1] proposed a non-uniform sampling scheme called Honeycomb, which allows sparse sampling at areas where the gradients of the cost function over position and direction are both small. The Honeycomb scheme first generates cells whose diameters are inversely proportional to a linear function of the two gradients. The sampling points are then put on the cell border. The efficiency of the Honeycomb scheme comes from its careful dependency on the cost function.

III. Shifted Paths

We modify the discrete path based on grid discretization and allow the end points of the segments to shift away from the grid points. The shifts are local, so the shifted path is a close variant to the original discrete path. With a shift the segments can take virtually any directions, thereby overcoming the limit-on-direction problem associated with grid sampling.

Recall the formulation of the path-oriented method in [1]. The compact region $\mathcal{R}$ is sampled by a finite set of points $\mathcal{X}$ which includes the starting point x_s. The discrete path $p(t)$ over $\mathcal{X}$ is defined to be a piecewise linear curve connecting a sequence of points in set $\mathcal{X}$ with respect to a given neighborhood distance δ. Specifically, there is a sequence of points

$$\{x_0 = x_s, x_1, x_2, ..., x_m = x\} \subset \mathcal{X} \tag{6}$$

that satisfy the neighborhood condition, as in (7) below,

$$||x_k - x_{k-1}|| \le \delta,\ \forall k \in \{1, 2, ..., m\}. \tag{7}$$

The discrete path $p(t)$ is defined as

$$p(t) = x_{k-1} + \frac{x_k - x_{k-1}}{||x_k - x_{k-1}||}(t - t_{k-1}), \quad \forall t \in [t_{k-1}, t_k], k \in \{1, 2, ..., m\}, \tag{8}$$

where t_k is defined recursively by

$$t_0 = 0, t_k = t_{k-1} + ||x_k - x_{k-1}||,\ \forall k \in \{1, 2, ..., m\}, \tag{9}$$

and $t_m = T$. The kth segment of $p(t)$, denoted by $\pi_{p,k}$, or π_k if there is no confusion, is the part of path $p(t)$ with $t \in [t_k, t_{k+1}]$, $k \in \{0, 1, ..., m-1\}$. Points x_k and x_{k+1} are called the start point and end point of segment π_k respectively. Note that $||\dot{p}(t) = 1||$ is assumed in the above definition, time is thus not important in most cases. It is then convinient to represent the discrete path p as $p = \{x_0 = x_s, x_1, ..., x_m = x\}$.

Henceforth we use grid sampling with grid size h to discretize the region. The discrete set $\mathcal{X}$ contains all points $x \in \mathcal{R}$ of the form $x = (x_{s,1} + i_1 h, ..., x_{s,n} + i_n h)$, where x_s is the starting point and $i_1, ..., i_n$ are integers. The *shifted path* can now be defined, starting with a discrete path $p = \{x_0 = x_s, x_1, ..., x_m\}$, in which the x_k's are grid points satisfying (7). A shifted path $\tilde{p}$ with respect to p is defined by the shift array $S = (s_1, ..., s_m)$ such that

$$\tilde{p}_S = \{\tilde{x}_0, \tilde{x}_1, ..., \tilde{x}_m\} = \{x_s, x_1 + s_1, ..., x_m + s_m\}, \tag{10}$$

$s_k \in \Omega$ are local shifts, and $\tilde{x}_k \in \Omega(x_k) := \{x_k + s : s \in \Omega\}$ is within the *local shift area* of x_k. One can apply definition (8) to obtain the shifted path $\tilde{p}_S(t)$ as

$$\tilde{p}_S(t) = \tilde{x}_{k-1} + \frac{\tilde{x}_k - \tilde{x}_{k-1}}{||\tilde{x}_k - \tilde{x}_{k-1}||}(t - \tilde{t}_{k-1}), \quad \forall t \in [\tilde{t}_{k-1}, \tilde{t}_k], k \in \{1, 2, ..., m\}, \tag{11}$$

where $\tilde{t}_k$ is defined recursively by

$$\tilde{t}_0 = 0, \tilde{t}_k = \tilde{t}_{k-1} + ||\tilde{x}_k - \tilde{x}_{k-1}||,\ \forall k \in \{1, 2, ..., m\}. \tag{12}$$

The local shift area Ω is the h-ball,

$$\Omega := \{s : s \in \mathbb{R}^n, ||s|| \le h\}. \tag{13}$$

The size of Ω matches the grid size h, assuring the locality of the shifts. Moreover, provided that the neighborhood distance δ is reasonably large, a segment of the shifted paths is able to take any directions as the end point of the segment shifts within the area of Ω, thus overcoming the limit-on-direction problem. This property is listed as the following proposition.

Proposition: Suppose the neighborhood distance $\delta \ge \sqrt{n}h$ where n is the dimension of the region and h is the discretization step. Consider a segment π with the start point $x \in \mathcal{X}$ and an arbitrary direction v ($v \in \mathbb{R}^n$, $||v|| = 1$ spans the surface of the unit ball). There exists a neighbor point $y \in \mathcal{Y}_x$ and a shift $s \in \Omega$ such that the end point of segment π becomes $\tilde{y} = y + s$, and the direction of π is same as the direction v. Specifically, there exists a scalar α such that

$$\tilde{y} - x = \alpha v. \tag{14}$$

We omit the technical proof. The proposition is intuitively obvious if $\tilde{y}$ spans a set that surrounds x completely, as does the surface of the unit ball centered at x. Fortunately, given $\delta \ge \sqrt{n}h$, $\mathcal{Y}_x$ contains all vertices of the cube of size h cornered at x. With Ω defined in (13), the set $\{\tilde{y} = y + s : y \in \mathcal{Y}_x, s \in \Omega\}$ contains the surface of this h-cube, therefore surrounds x completely, and thereby eliminates the limit-on-direction problem of the grid discretization. Though the definition of Ω in (13) is sufficient for this

purpose, the set may be too large. We can take as Ω the low-dimensional set that restricts the shift s to be perpendicular to the path,

$$\Omega := \{s : s \in \mathbb{R}^{n-1}, ||s|| \leq h, \langle s, \dot{p} \rangle = 0\}, \quad (15)$$

and still overcome the limit-on-direction problem. Taking Ω as in (15) reduces the number of parameters associated with a shift path so that the optimization procedure described below remains simple.

The concept of shifted paths gives a new meaning to the value function of a discrete path in the context of grid discretization. Given a discrete path $p = \{x_0 = x_s, x_1, ..., x_m = x\}$, the *tilde value function* $\tilde{V}(x,p)$ is defined by

$$\tilde{V}(x,p) = \min_{s_k \in \Omega, \forall 1 \leq k < m} V(x, \tilde{p}_S). \quad (16)$$

The minimization is over all local shifts of the discrete path p, so $\tilde{V}(x,p)$ is the lowest value one can expect by choosing p. Therefore, even when the grid is coarse, the minimizing shifted path $\tilde{p}$ has a good chance to closely approximate a shortest continuous path. As we refine the grid, the optimal $\tilde{V}(x,p)$ will converge to the optimal solution of Problem (2).

Theorem: For any $A, \epsilon > 0$, there exists a finite set $\mathcal{X}$ of N_ϵ grid sampled points of $\mathcal{R}$, due to the discretization step h, such that for any given starting point $x_s \in \mathcal{X}$ and any $x \in \mathcal{X}$,

$$\min_{p \in P_{\mathcal{X}}(x_s,x)} \tilde{V}(x,p) - \epsilon \leq V(x) \leq \min_{p \in P_{\mathcal{X}}(x_s,x)} \tilde{V}(x,p), \quad (17)$$

where $P_{\mathcal{X}}(x_s, x)$ contains all the discrete paths from x_s to x over $\mathcal{X}$, A is the bound on the second derivative of the continuous path p, and $V(x)$ is the value function of the shortest path of the Anisotropic Shortest Path problem, as defined in (2).

The theorem is proved along along the same lines as Theorem 1 in [1]. The latter guarantees the existence of a finite set $\mathcal{X}$ of size N_ϵ such that one can construct a discrete path p_0 over $\mathcal{X}$ satisfying

$$V(x, p_0) - V(x) \leq \epsilon. \quad (18)$$

Notice that the definition of the tilde value function in (16) implies $\tilde{V}(x, p_0) \leq V(x, p_0)$, so the convergence theorem holds immediately. However, this argument doesn't reveal how fast the number of points N_ϵ increases, or equivalently, how fast h decreases, as ϵ decreases. In the numerical experiments the shifted paths perform much better than the discrete paths, meaning that the proposed algorithm can work over a coarse grid with a small number of points in $\mathcal{X}$, and yet yield paths with values close to the optimum.

IV. Algorithm

The underlying scheme of the proposed Shiftable Segment-by-Segment Pathing (SSSP) algorithm is the Bellman-Ford architechture, in which each point records the most promising paths and their tilde value functions

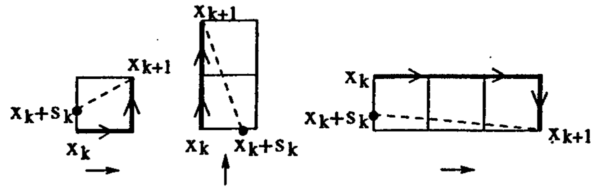

Fig. 4. Some examples of segments (thick lines with arrow) and their baseline directions (below) when $l = 3$. s_k is the shift along the grid line that is perpendicular to the baseline direction. The dashed lines show the shifted segments.

from the starting point and tries to extend them to the destination through its neighbors. The paths eventually reach the destination and the shortest path (with the smallest tilde value) among them is the solution. Note that the recorded paths are discrete paths. To find the tilde values of these paths, one has to solve an optimization problem over the shift vector s_k for all segments π_k. If a path p has m segments, it is an optimization problem over Ω^m, which is obviously complicated and time consuming. We introduce a heuristic method to compute the tilde values approximately. From now on we restrict the region to a two-dimensional plane. According to (15), the shifts s_k's are now scalars.

Consider the graph formed by a rectangular grid with only vertical and horizontal links between adjacent points. Let the segment π be a group of links of up to l base links and 0 or 1 perpendicular link, as shown in Fig. 4. When $l = 1$, it corresponds to the case where the neighborhood distance $\delta = \sqrt{2}h$; and $l = 3$ implies $\delta = \sqrt{10}h$. The direction of the base links is called the *baseline direction* of the segment, while the direction the perpendicular link, if there is one, is called the *quadrantal direction*. The shift s_k of the start point of a segment π_k is along the grid line that is perpendicular to the baseline direction of π_k. The assumption here is that the baseline direction represents the direction of the path.

The main loop of the SSSP algorithm follows the Bellman-Ford architecture. For a graph of $n = |\mathcal{X}|$ points, the architecture does up to $n-1$ rounds over all links relaxing, or updating, the tilde value function (the distance to the starting point x_s), as well as the shifted path that generates it. This information forms a record of a point $x \in \mathcal{X}$, including the discrete path p_x^*, the shifts S_x^*, and the corresponding tilde value function $\tilde{V}^*(x, p_x^*)$. In practice, the relaxation operation is initiated by a recently updated point x_k when, from a path-oriented view, its recorded path $p_k := p_{x_k}^* = \{x_s, x_1, ..., x_{k-1}, x_k\}$ tries to extend to a neighbor point x_{k+1}. Denote the extended path by $p_{k+1} := \{p_k, x_{k+1}\}$; the records at x_{k+1} will be relaxed, or updated, if the tilde value of p_{k+1} is smaller than the recorded $\tilde{V}^*(x_{k+1}, p_{x_{k+1}}^*)$. Initialized to infinity, the tilde value function therefore decreases with each successful relaxation operation. The following procedures describe the details of the relaxation operation of SSSP.

1) Approximate the tilde value function $\tilde{V}(z_k, p_k)$ of the

point z_k that shifts away from the grid point x_k. $z_k = x_k + s'_k e_k$ is the shifted point, $s'_k \in \Omega$ is the shift, and e_k is the unit vector representing the direction that is perpendicular to the baseline direction of the segment $\pi_k = (x_k, x_{k+1})$. This is the key computation of SSSP and we will discuss it in detail soon.
2) Compute the tilde value function $\tilde{V}(x_{k+1}, p_{k+1})$ of the extended path p_{k+1} as

$$\tilde{V}(x_{k+1}, p_{k+1}) = \min_{s'_k}(\tilde{V}(z_k, p_k) + ||x_{k+1} - z_k|| \cdot c(x_{k+1}, x_{k+1} - z_k)), \quad (19)$$

where $c(x, v)$ is the cost function.
3) If $\tilde{V}(x_{k+1}, p_{k+1}) < \tilde{V}^*(x_{k+1}, p^*_{x_{k+1}})$, the relaxation operation succeeds and the record of x_{k+1} is updated.

The Bellman-Ford architecture of SSSP allows each grid point to keep one record of the most promising path. However, because of the failure of the Principle of Optimality, a sub-optimal path at x_k may extend to be the best path at x_{k+1}. Therefore, it is efficient to keep multiple records of the most promising paths at each points. The size of the records doesn't need to be large. For example, when the number of base links of a segment is selected to be $l = 1$, there are at most eight different choices of neighbor points. Therefore, each point needs to keep no more than eight paths, more paths make no difference. Moreover, keeping multiple records increases the time complexity of the algorithm. If each point keeps q records, the Bellman-Ford architecture will need up to $q(n-1)$ rounds of relaxation operations, so a small q is better. (In our implementation, we choose $q = 4$.)

The bound of the second derivative of the path, A, enters in the following way. Due to the assumption of constant speed, the derivative of the speed comes from a direction change. Obviously, a piecewise linear trajectory has unbounded second-order derivative. We therefore limit the impact of the action of changing directions: the action doesn't finish all at once; instead, it takes a small amount of time proportional to the time needed to traverse distance h. The assumption guarantees that there is enough room to finish the action of changing directions. Also, fine discretization reduces the capability to changing directions from one segment to another. As $||\dot{p} = 1||$, the angle ϕ between two consecutive segments is limited by

$$\phi \leq \arccos\left(1 - \frac{hA}{2}\right). \quad (20)$$

Returning to step 2 of the relaxation operation, only if the angle between the new segment π_k and the last segment of the path p_k satisfies this inequality, is the extension feasible. The corresponding shift s'_k is called a feasible shift.

The key step in the SSSP algorithm is the computation of the approximate tilde value function $\tilde{V}(z_k, p_k)$ at the shifted point z_k in step 1. In the isotropic case this can be done by interpolation, as in the Fast Marching algorithm.

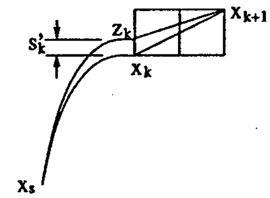

Fig. 5. Extension of a shifted path. When considering the next segment $\pi_k = (x_k, x_{k+1})$, p_k shifts z_k in a shape-preserving way.

What we want here, however, is a path-oriented method that takes as an input only the shifted path $\tilde{p}_{k,S_k}$, given by the recorded discrete path p_k and shifts S_k. The idea, roughly speaking, is to shift the path in a "shape-preserving" way, as shown in Fig. 5. When the end point of the path $\tilde{p}_k$ shifts from x_k to a non-grid point $z_k = x_k + s'_k e_k$, the trajectory of the path should shrink (or swell) in such a way that the resulting path keeps its original shape. By doing so, the directions of all the segments of the path are almost the same as those before the shift. We call such shifts *shape-preserving*. The shape-preserving shift results in a backward propagation that affects the shifts of all the segments, turning shifts $S_k = (s_1, ..., s_{k-1}, 0)$ into $S_{k+1} = (s_1 + s'_1, ..., s_{k-1} + s'_{k-1}, s'_k)$. The value function over the shape-preserving shifted path $\tilde{p}_{k,S_{k+1}}$ is thus a good approximation of the tilde value function $\tilde{V}(z_k, p_k)$. The collection of these values then forms a profile that is used to compute the minimizing shift s'_k in step 2.

It is not necessary to complete the backward propagation before computing the value functions of the shape-preserving shifted paths. We can do it recursively based on the knowledge of the profile of $\tilde{V}(z_{k-1}, p_{k-1})$. Let s'_{k-1} be the first-step propagation due to the shape-preserving shift s'_k. We have $z_{k-1} = x_{k-1} + (s_{k-1} + s'_{k-1})e_{k-1}$ and $z_k = x_k + s'_k e_k$. Thus,

$$\tilde{V}(z_k, p_k) = \tilde{V}(z_{k-1}, p_{k-1}) + ||z_k - z_{k-1}|| \cdot c(x_k, z_k - z_{k-1}). \quad (21)$$

Based on the property of the shape-preserving shift, the propagated shift s'_{k-1} depends on s'_k in one of two ways. The first is the proportional shift. It occurs when segments π_{k-1} and π_k have the same baseline direction and the same quadrantal direction. With proportional shift, $s'_{k-1} = \frac{x_{k-1} \cdot e_{k-1}}{x_k \cdot e_k} s'_k$. It is the consequence of the fact that x_s, the start point of the first segment, never shifts. If the path turns, either by changing the baseline direction or the quadrantal direction, the propagated shift happens in a parallel manner. In this case the last segment π_{k-1} retains its direction and s'_{k-1} is computed accordingly.

With the profile of the approximate tilde value function $\tilde{V}(z_k, p_k)$, the optimization problem in step 2 becomes a minimizing problem over a single scalar variable s'_k, thus is significantly simplified. The profile is in a piecewise linear form over the inteval $s'_k \in \Omega = [-1, 1]$. However, not every $s'_k \in [-1, 1]$ is feasible even if it satisfies the direction condition (20). In the example shown in Fig. 6,

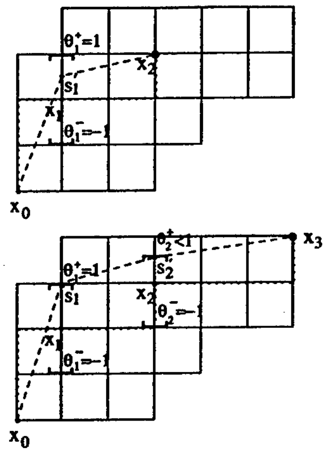

Fig. 6. θ_k^+/θ_k^- bounds of the shifts. The dotted lines show how the path extends in the gird graph; the dashed lines show the shift paths; and the brackets show the θ_k^+/θ_k^- bounds of the shifts. In the upper figure, θ_1^+/θ_1^- spans the $[-1,1]$ interval. When the path extends to x_3, as shown in the lower figure, s_2' is bounded by $\theta_2^+ < 1$ from above. This is because the shift s_1 of the extended path increases in the shape-preserving way, and at the same time has to be less than or equal to 1.

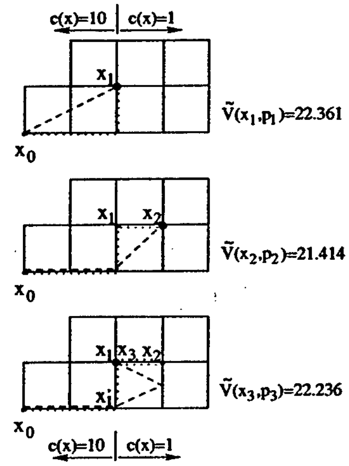

Fig. 7. This figure shows how SSSP generates a loop when the path extends from x_1 to x_2, then back to $x_3 = x_1$. The dotted lines show how the path extends in the grid graph; the dashed lines are the shifted paths. The reason of generating loops is that the tilde value function decreases as the paths extend, as shown on the right.

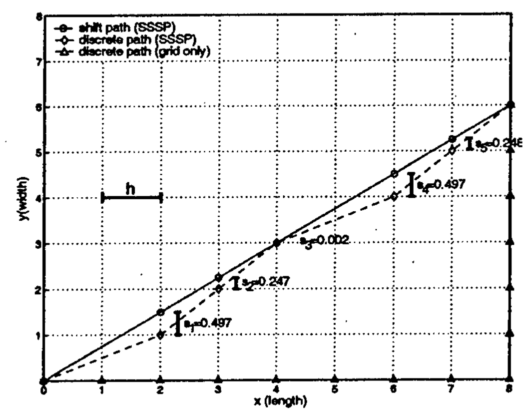

Fig. 8. Path computation problem over a rectangular region with flat cost function $c(x,v) = 0.1$. The grid in dotted line shows the grid discretization of the region. s_k's are the shifts. The triangle-labeled discrete path shows how serious the limit-on-direction problem could be.

s_2' is bounded by $\theta_2^+ < 1$ because s_1 after the backward propagation must be within its $[-1,1]$ interval. We denote these bounds θ_k^+/θ_k^-. If these bounds are tight, it can be proven that the resulting shifted path has all shifts less than or equal to 1.

The SSSP algorithm is *not* loop-free. Although the cost function $c(x,v)$ is strictly positive at all x and v, the tilde value function may decrease as the path extends. See Fig. 7 for reference. Suppose the (isotropic) cost function takes only two values: $c(x) = 10$ for all points to the left of x_1, and $c(x) = 1$ for all points to the right. We can compute the tilde value functions $\tilde{V}(x_1,p_1) = \sqrt{5} \times 10 = 22.361$, and $\tilde{V}(x_2,p_2) = 2 \times 10 + \sqrt{2} = 21.414$. Therefore, by extending the path p_1 to x_2, the tilde value decreases by 0.947. The decreasing tilde value implies the possibility of loops. In fact, remember that SSSP records multiple paths at each point. The multiple records also imply potential loops as paths extend. We definitely don't want looped paths. One way of preveting loops is to explicitly check the paths and delete those with loops. However, the looped path is inherent within the shifted path context and makes sense. In the lowest part of Fig. 7, when we extend the path back to $x_3 = x_1$, the tilde value function of that point improves to $\tilde{V}(x_3,p_3) = 2\times 10 + \sqrt{5} = 22.236$, based on the path $\tilde{p}_3$ with the dashed line. Therefore, though p_3 contains loop, the shifted path $\tilde{p}_3$ doesn't. Fortunately, $\tilde{p}_3$ is still achievable by replacing x_1 with x_1'. Thus, the explicit loop elimination doesn't affect the optimality of the results.

Unlike random sampling algorithms, which use up running time in constructing the graph, the SSSP algorithm spends more time on its path computation. Specifically, for a graph of n sample points and m links, random sampling algorithms take $O(n^2)$ time to select links by checking the distance between the sample points, and $O(m)$ time to compute the values of each link. The SSSP algorithm, on the other hand, needs $O(qmn)$ relaxation operations, where q is the number of records kept at each point of the graph. During each of the relaxation operation, SSSP performs a fixed number of computations to form a profile of the tilde value function $\tilde{V}(z,p)$, and to find the minimizing shift s'. The time complexity of the SSSP algorithm is thus $O(qmn)$. This is a similar expression as the time complexity of the random sampling algorithms. However, SSSP can find good paths even with a coarse grid and fewer points in the graph. Therefore, the running time of SSSP should be smaller than that of the random sampling algorithms, as we show in the next section.

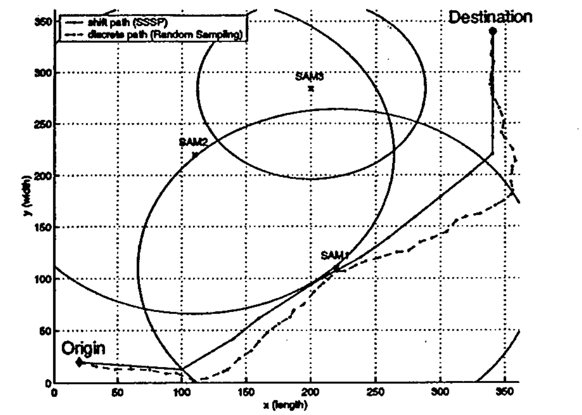

Fig. 9. The typical senario and minimum-risk paths computed by two algorithms. (i). the proposed SSSP algorithm: 361 (19×19) points, path value is 358.2, running time is 2.6 sec; (ii). the uniform random sampling algorithm: 3000 points, path value is 375.3, running time is 74.6 sec.

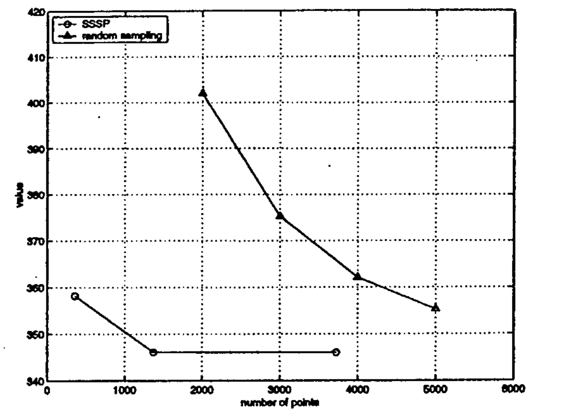

Fig. 10. Results value functions of the SSSP algorithm and random sampling algorithm at different sampling rates.

V. NUMERICAL RESULTS

The first set of experiments illustrates the effect of the shift concept. Fig. 8 shows a senario where the cost function is 0.1 at all positions and directions. The flat cost function represents the simplest case of the path computation problem. The simple example demonstrates how the shiftment helps to find better paths. In Fig. 8, an 8-by-8 region is discretized with grid size $h = 1$. We want to compute the shortest path from the starting point $x_s = (0,0)$ to the destination point $x_d = (8,6)$. If we run Dijkstra's shortest path algorithm over the grid graph, the triangle-labeled discrete path is the solution with path value being 1.4. (Actually, all the zigzag paths from x_s to x_d are equally the shortest path.) This obviously bad solution can be improved by increasing the neighborhood distance, which allows longer segments than those only between the adjacent points on grids. The improved discrete path is represented by the dashed line and diamond label in Fig. 8. The path value reduces to 1.013. The discrete path cannot achieve the optimum value 1.0 of the straight line connecting the starting and destination points, because of the "limit-on-direction" problem mentioned before. By introducing the shifted path, the segments are free to take different directions, thus generate the near-to-optimal path. The shifted path, labeled by circles with path value 1.0, is very close to the optimal straight line. We also display in Fig. 8 the amount of the shifts at the consecutive grid points of the discrete path.

We now apply the proposed SSSP algorithm to the computation of paths for Unmanned Air Vehicles (UAVs) that minimize the risk of detection (and destruction) by ground radars of Surface-to-Air Missile (SAM) sites. Following [1], finding the minimum risk path is abstracted as the ASP problem with cost function

$$c(x, v) = \sum_i \eta(x, v, z_i), \tag{22}$$

where z_i is the position of the ith radar, and

$$\eta(x, v, z_i) = \frac{ab}{\sqrt{a^2 + (b^2 - a^2)\cos^2\psi}} \tag{23}$$

is the risk density function, representing the probability of the UAV being detected by the ith SAM site. This probability depends on the aircraft's Radar Cross Section (RCS), which is a measure of its ability to reflect radar signals in the direction of the radar receiver. In (23), the RCS is an ellipse with semi-minor axis a and semi-major axis b. Unlike most civilian aircrafts whose RCS is close to a circle, the UAVs have flat RCS with large $r = b/a$. Therefore, if ψ, the angle between the moving direction v and the SAM site direction $(x - z_i)$, $\eta(x, v, z_i)$ takes the small value a if $\psi = 0$ and the large value b when $\psi = 90$ degrees.

Fig. 9 considers the senario with two long range SAM sites and one short range SAM. The circles show the effective ranges of the SAM radars. The figure also shows the minimum-risk paths obtained by the SSSP algorithm and the uniform random sampling algorithm, respectively. The SSSP algorithm runs over a 19×19 grid, or 361 points and yields a shifted path with tilde value 358.2. The uniform random sampling algorithm, at the same time, runs over 3000 randomly sampled points and yields a discrete path of value 375.3. Moreover, the running time of latter is nearly 30 times slower than that of SSSP. The next two figures, Fig. 10 and 11, show more results of the two algorithms as we change the sampling rates. The SSSP algorithm keeps outperforming the uniform random sampling algorithm on both the path values and the running time. They also demonstrate that SSSP is able to achieve the optimal solution to the ASP problem with a very low sampling rate.

VI. CONCLUSION

We studied the problem of computing the shortest paths over a continuous region in this paper. The cost depends on both the position and direction of movement, thus forming an Anisotropic Shortest Pathing (ASP) problem. We

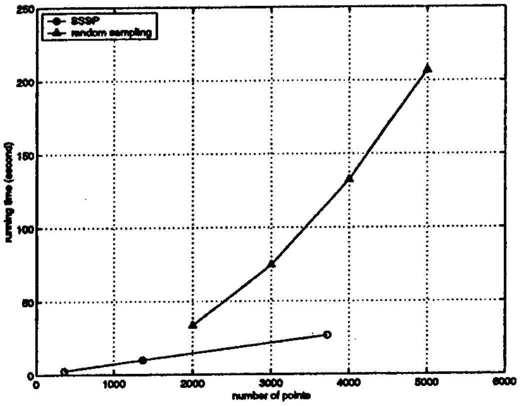

Fig. 11. Results running time of the SSSP algorithm and random sampling algorithm at different sampling rates.

proposed a Shiftable Segment-by-Segment Pathing (SSSP) algorithm that has the following features.

First, SSSP uses rectangular grid to discretize the region. Compared with other sampling schemes, the grid graph is easy to construct and is independent of the cost function. Second, the SSSP algorithm uses a Bellman-Ford architecture to perform a breadth-first search. Each point of the graph keeps multiple records of most promising paths to assure the optimality of the result. The third feature is the introduction of shifted paths, which allow the discrete paths to shift away from the grid points, thereby overcoming the limit-on-direction problem that arises in grid discretization. The last feature is the heuristic to compute the tilde value function $\tilde{V}(x,p)$. By constructing the profile of the tilde value functions $\tilde{V}(z,p)$ over the shifted points z, we replace the optimization problem over all shifts to the simpler one over only the last shift of the path. The key technique of the heuristic is the shape-preserving shift, which decides how the shifted path changes when the end point shifts.

These features of SSSP work together and lead to an efficient algorithm. The theoretical analysis says that the SSSP algorithm has a similar time complexity as the random sampling algorithms. The numerical experiments, on the other hand, show that SSSP outperforms the uniform random sampling algorithm on both the path values and the running time.

Although we described and implemented the SSSP algorithm for two-dimensional regions, it is easy to generalize the algorithm to higher dimensional spaces. The major modifications focus on the local shift area Ω, the shifts s_k's and the computation of the profile of $\tilde{V}(z,p)$. For example, in a three-dimensional space, the shift is restricted within a small two dimensional plane, $\Omega(x)$, around the grid point x. Of course, the algorithm will need more computations to construct the profile of $\tilde{V}(z,p)$ over $\Omega(x)$.

REFERENCES

[1] J. Kim and J. Hespanha "Discrete Approximations to Continuous Shortest-Path: Application to Minimum-Risk Path Planning for Groups of UAVs", *Proc. of the 42th Conf. on Decision and Contr*, pp.936-46, 2003.

[2] D. Adalsteinsson and J.A. Sethian, "A fast level set method for propagating interfaces", *Journal of Computational Physics*, 118, 269-277, 1995.

[3] J.A. Sethian and A. Vladimirsky "Ordered upwind methods for static Hamilton-Jacobi equations", *Technical Report PAM-792*, Center for Pure and Applied Mathematics, University of California at Berkeley, 2001.

[4] J.N. Tsitsiklis "Efficient algorithm for globally optimal trajectories", *IEEE Transactions on Automatic Control*, 40(9), 1528C1538, Sep 1995.